### CYK Algorithm: More Details

- **S → AB**
- **A → AA | AB | a**
- **B → CC**
- **C → b**
- **x = aaabb (n = 5)**

Start by filling in the “top” row in the table, with the variables that directly produce the terminal symbol at the corresponding position.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

E.g. the single symbol at position 5 in the string is directly produced by variable C, so we put a C in the entry at column 5, row 1.
### CYK Algorithm: More Details

- $S \rightarrow AB$
- $A \rightarrow AA \mid AB \mid a$
- $B \rightarrow CC$
- $C \rightarrow b$
- $x = aaabb \ (n = 5)$

Fill in subsequent rows by looking at the cells directly above and the cells diagonally to the upper right. (How to do this isn’t very clear using row 2 as an example, so I’ll wait for row 3 to go over it in more detail.)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$A$</td>
<td>$A$</td>
<td>$A$</td>
<td>$C$</td>
<td>$C$</td>
</tr>
<tr>
<td>2</td>
<td>$A$</td>
<td>$A$</td>
<td>$\emptyset$</td>
<td>$B$</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fill in subsequent rows by looking at the cells directly above and the cells diagonally to the upper right. (How to do this isn’t very clear using row 2 as an example, so I’ll wait for row 3 to go over it in more detail.)
### CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow AA | AB | a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]

\[ x = aaabb \ (n = 5) \]

Say we want to fill in this cell next.
CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

We want to fill the red cell with the set of variables that produce the substring with length \( j \) that starts at position \( i \) (in this case, we want variables that produce \( aaa \)).
We are going to look for ways to produce a substring of length 3 starting at position 1 by concatenating two shorter strings. We know that the first of these shorter strings must start at the same position as the new, longer substring.

Substrings starting at position 1 are produced by variables in this column.
### CYK Algorithm: More Details

We know that, in order to have a substring of length 3, we need to choose the second of our two strings in such a way that the length of the combined string is 3.

**Example:**

Given the string $x = aaabb$ ($n = 5$), we need to find a substring of length 3. We can achieve this by choosing the second string in such a way that the combined string's length is 3.

### Table Representation

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td>A</td>
<td>A</td>
<td>$\emptyset$</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Diagonal Moves**

- A string produced by a variable in cell 1,1 has a length of 1.

**Calculation**

- So, to get a string of length 3, we need to concatenate the first part with a second part that starts at position 2 and is of length 2. Such strings are produced by variables in cell 2,2 of the table.
**CYK Algorithm: More Details**

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

So, now we know that we can get a substring of length 3 starting at position 1 by concatenating two strings that are produced by variable \( A \). So, we need to look for productions that produce two \( A \)'s concatenated together.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>j</td>
<td>A</td>
<td>A</td>
<td>∅</td>
<td>B</td>
<td></td>
</tr>
</tbody>
</table>

So, now we know that we can get a substring of length 3 starting at position 1 by concatenating two strings that are produced by variable \( A \). So, we need to look for productions that produce two \( A \)'s concatenated together.
In this case, one such production exists: $A \rightarrow AA$. We will therefore add the “producer” variable to the list of variables in cell 1,3.
CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

We can continue checking all of the possible ways to produce strings of length 3 starting at position 1. However, in this case, we’re just looking at two A’s again so it’s not very interesting.

A string produced by a variable in cell 1,2 has a length of 2.

So, to get a string of length 3, we need a string that starts at position 3 and is of length 1. Such strings are produced by variables in cell 3,1 of the table.
CYK Algorithm: More Details

Let’s do the same exercise for the next cell in the table.

\[
S \rightarrow AB
\]
\[
A \rightarrow AA | AB | a
\]
\[
B \rightarrow CC
\]
\[
C \rightarrow b
\]
\[
x = aaabb \ (n = 5)
\]
### CYK Algorithm: More Details

S → AB
A → AA | AB | a
B → CC
C → b

\[ x = aaabb \ (n = 5) \]

In this case, we want strings starting at position 2 of length 3.

\[ x = aaabb \ (n = 5) \]

<table>
<thead>
<tr>
<th></th>
<th>i →</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>A</td>
<td></td>
<td></td>
<td>B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A string produced by a variable in cell 2,1 starts at position 2 and has length 1.

So, to get a string of length 3, we need a string that starts at position 3 and is of length 2. Such strings are produced by variables in cell 3,2 of the table.
CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

No possible productions produce a variable in cell 2,1 followed by a variable in cell 3,2, simply because cell 3,2 doesn’t contain any variables.
## CYK Algorithm: More Details

The CYK algorithm is used to parse a string and determine if it is generated by a given context-free grammar. Let's consider the following grammar:

\[
\begin{align*}
S & \rightarrow AB \\
A & \rightarrow AA \mid AB \mid a \\
B & \rightarrow CC \\
C & \rightarrow b
\end{align*}
\]

And the string we are trying to parse is \(x = aaabb\) (\(n = 5\)).

### Productions

- \(S \rightarrow AB\)
- \(A \rightarrow AA \mid AB \mid a\)
- \(B \rightarrow CC\)
- \(C \rightarrow b\)

### Parsing Table

Now, we are looking for productions that produce an \(A\) followed by a \(C\). However, there aren't any.

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

A string produced by a variable in cell 2,2 starts at position 2 and has length 2.

So, to get a string of length 3, we need a string that starts at position 4 and is of length 1. Such strings are produced by variables in cell 4,1 of the table.
CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

We weren’t able to find any productions for this cell, so we’ll just indicate that the “set of variables producing substrings of length 3 starting at position 2” is empty.

We have the following productions:

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]

For the string \( x = aaabb \) (\( n = 5 \)), the CYK algorithm table looks like this:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>b</td>
<td>b</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>A</td>
<td>A</td>
<td>\emptyset</td>
<td>B</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>A</td>
<td>\emptyset</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Following the same pattern, for cell 3,3, we’ll first look for productions that produce an $A$ followed by a $B$.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>i</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>j</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td>A</td>
<td>A</td>
<td>φ</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>A</td>
<td>φ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CYK Algorithm: More Details

There are two variables that produce this: $S$, and $A$. So, we’ll put both of these variables into the set in cell 3,3.

<table>
<thead>
<tr>
<th>i</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>j</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>$A$</td>
<td>$A$</td>
<td>$\textcolor{blue}{A}$</td>
<td>$C$</td>
</tr>
<tr>
<td>2</td>
<td>$A$</td>
<td>$A$</td>
<td>$\emptyset$</td>
<td>$\textcolor{blue}{B}$</td>
</tr>
<tr>
<td>3</td>
<td>$A$</td>
<td>$\emptyset$</td>
<td>$S,A$</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The next entries we check once again involves an empty cell, so we don’t add anything else to cell 3,3.
Now, we’ll move on to cell 4,1, corresponding to substrings of length 4 starting at position 1.

\[
S \rightarrow AB \\
A \rightarrow AA \mid AB \mid a \\
B \rightarrow CC \\
C \rightarrow b \\
x = aaabb \ (n = 5)
\]
First, we’ll check these two cells, one of which is empty so we clearly won’t find any productions for this combination.

Reminder: produces substrings of length 1 starting at position 1.

Produces substrings of length 3 starting at position 2 (basically this is saying that no variable in this grammar can produce the string $aab$.)
CYK Algorithm: More Details

\[
S \rightarrow AB \\
A \rightarrow AA \mid AB \mid a \\
B \rightarrow CC \\
C \rightarrow b \\
x = aaabb \ (n = 5)
\]

Next, there’s another production involving an empty set, so we won’t find anything here.
CYK Algorithm: More Details

Finally, we’ll check for productions that produce an $A$ followed by a $C$, and there aren’t any.

$x = aaabb$ ($n = 5$)

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>$A$</td>
<td>$A$</td>
<td>$A$</td>
<td>$C$</td>
<td>$C$</td>
</tr>
<tr>
<td>2</td>
<td>$A$</td>
<td>$A$</td>
<td>$\emptyset$</td>
<td>$B$</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>$A$</td>
<td>$\emptyset$</td>
<td>$S,A$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CYK Algorithm: More Details

So, cell 1,4 is empty.

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

\[
\begin{array}{cccccc}
 & 1 & 2 & 3 & 4 & 5 \\
1 & A & A & A & C & C \\
2 & A & A & \emptyset & B \\
3 & A & \emptyset & S,A \\
4 & \emptyset \\
5 & \\
\end{array}
\]

Jim Anderson (modified by Nathan Otterness)
Next, we’ll check cell 2,4. The first combination we check has two possible variables corresponding to the second part of the substring, so we’ll look for productions of $AS$ and $AA$ (using both possible variables in the second cell).
CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow [AA] AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

Nothing produces an \( S \), however \( A \) produces \( AA \), so we can add \( A \) to cell 2,4.

\[
\begin{array}{cccccc}
\text{a} & \text{i} \rightarrow & \text{a} & \text{a} & \text{b} & \text{b} \\
1 & 2 & 3 & 4 & 5 \\
\hline
\text{1} & A & [A] & A & C & C \\
\text{2} & A & A & \emptyset & B \\
\text{3} & A & \emptyset & [S,A] \\
\text{4} & \emptyset & A \\
\end{array}
\]

Jim Anderson (modified by Nathan Otterness)
### CYK Algorithm: More Details

\[
S \rightarrow [AB]
\]
\[
A \rightarrow AA \ | [AB] \ a
\]
\[
B \rightarrow CC
\]
\[
C \rightarrow b
\]
\[
x = aaabb \ (n = 5)
\]

Next, we’ll look for productions that produce \(AB\), and we see that \(S\) and \(A\) produce \(AB\). Cell 2,4 already contains \(A\), so we only need to add the \(S\).

\[
\begin{array}{cccccc}
\hline
 & a & a & a & b & b \\
\hline
1 & i \rightarrow & 1 & 2 & 3 & 4 & 5 \\
3 & A & A & \emptyset & B \\
4 & \emptyset & S, A \\
5 & \emptyset & S, A \\
\hline
\end{array}
\]
**CYK Algorithm: More Details**

\[
\begin{align*}
S & \to AB \\
A & \to AA | AB | a \\
B & \to CC \\
C & \to b \\
x & = aaabb \ (n = 5)
\end{align*}
\]

Finally, we know we won’t get any new productions from these two cells, so we’re done with cell 2,4.

![Table and Diagram]

Jim Anderson (modified by Nathan Otterness)
Finally, we need to fill in the last cell, corresponding to a string starting at position 1 of length 5. (In other words, the entire string.) We’ll just work through the possibilities like before.

\[
\begin{array}{cccccc}
  & a & a & a & b & b \\
 1 & 2 & 3 & 4 & 5 \\
\end{array}
\]

\[
\begin{array}{cccccc}
 1 & 2 & 3 & 4 & 5 \\
\end{array}
\]

\[
\begin{array}{cccccc}
  & A & A & \emptyset & B \\
 2 & 3 & 4 & 5 \\
\end{array}
\]

\[
\begin{array}{cccccc}
  & A & \emptyset & S,A \\
 3 & 4 & 5 \\
\end{array}
\]

\[
\begin{array}{cccccc}
  & \emptyset & S,A \\
 4 & 5 \\
\end{array}
\]

\[
\begin{array}{cccccc}
  & \emptyset & S,A \\
 5 \\
\end{array}
\]
Once again, we’re looking for productions that produce either $AS$ or $AA$. $A$ produces $AA$, so add it to cell 1,5.

\[
\begin{array}{cccccc}
  & a & a & a & b & b \\
 1 & A & A & A & C & C \\
 2 & A & A & \emptyset & B \\
 3 & A & \emptyset & S,A \\
 4 & \emptyset & S,A \\
 5 & A \\
\end{array}
\]

$S \rightarrow AB$

$A \rightarrow [AA] AB \mid a$

$B \rightarrow CC$

$C \rightarrow b$

$x = aaabb \ (n = 5)$
In this example, we ended up already considering the next combination, so we’ve already added the variables we need to.
## CYK Algorithm: More Details

The CYK algorithm is a dynamic programming algorithm used for parsing context-free grammars. It builds a table that stores the set of non-terminals that could produce each substring of the input string.

### Grammar Rules
- \( S \rightarrow AB \)
- \( A \rightarrow AA \mid AB \mid a \)
- \( B \rightarrow CC \)
- \( C \rightarrow b \)

### Input String
- \( x = aaabb \) (\( n = 5 \))

### Algorithm Steps
1. **Initialization**
   - Start with the input string: \( x = aaabb \)
   - Create a table with rows and columns corresponding to the indices of the string.

2. **Fill the Table**
   - For each substring of length 1 to \( n \)
   - Check if the substring matches any of the productions.
   - If it matches, mark the cell with the non-terminal.

3. **Find the Production for \( AB \)**
   - We next look for productions producing \( AB \).
   - We already added \( A \) to cell \( i=1, j=5 \), so we only need to add the \( S \).

### Table Example

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>a</th>
<th>a</th>
<th>b</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>A</td>
<td>A</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td>A</td>
<td>A</td>
<td>∅</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>A</td>
<td>∅</td>
<td>S,A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>∅</td>
<td>S,A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>S,A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Jim Anderson (modified by Nathan Otterness)
CYK Algorithm: More Details

\[ S \to AB \]
\[ A \to AA | AB | a \]
\[ B \to CC \]
\[ C \to b \]
\[ x = aaabb \ (n = 5) \]

And finally, we have a combination involving an empty cell—no productions from this.

\[
\begin{array}{cccccc}
  & a & a & a & b & b \\
1 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & A & A & A & C & C \\
2 & A & A & \emptyset & B & \\
3 & A & \emptyset & S,A & \\
4 & \emptyset & S,A & \\
5 & S,A & \\
\end{array}
\]
CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow AA \mid AB \mid a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]
\[ x = aaabb \ (n = 5) \]

So now we’re finished filling out the table. To know if the string is in the language produced by the CFG, we only need to see if the start symbol is in the last cell.

\[
\begin{array}{cccccc}
  & a & a & a & b & b \\
 1 & 1 & 2 & 3 & 4 & 5 \\
\end{array}
\]

\[
\begin{array}{cccc}
  i & 1 & A & A & A \\
  j & 2 & A & A & \emptyset \\
  3 & A & \emptyset & S,A \\
  4 & \emptyset & S,A \\
  5 & S,A \\
\end{array}
\]
CYK Algorithm: More Details

\[ S \rightarrow AB \]
\[ A \rightarrow AA | AB | a \]
\[ B \rightarrow CC \]
\[ C \rightarrow b \]

\[ x = aaabb \; (n = 5) \]

In this case, cell 1,5 does contain the start symbol \( S \), so the string \( aaabb \) is in the language produced by the grammar.

Recall that this is saying that the start symbol is able to produce the “substrings” with a length of 5 starting at position 1.